Fundamentals of Computer Organization and Architecture

Solution Manual

Mostafa Abd-El-Barr and Hesham El-Rewini

September 2005

Chapter (1)

1-

Attribute	Trend
Cost of Hardware	Decreasing
Size of Memory	Increasing
Speed of Hardware	Increasing
Number of Processing Elements	Increasing
Geographical locations of system components	Increasing

2- It is expected that computing will continue to be fast, distributed, and multiprocessing-based

3-

a) Cluster Computing:
b) Grid Computing:
c) Quantum Computing:
d) Nanotechnology:

4- A clock rate of 1GHz corresponds to 1 nsec processing time. For a speed of 300,000 Km/sec, the maximum distance should be $300{,}000 \times 100000 \times 10^{-9} = 30$ cm !!. In case that the clock rate is 1THz, the processing time will be 1 pico-second. In the latter case, the maximum distance will be $300{,}000 \times 100000 \times 10^{-12} = 0.030$ cm = 0.3 mm!!

5-

Aspect	Uniprocessor	Multiprocessor
Ease of Programming	Feasible	Involved
Nedd for Synchronization	NA	Essential
Performance evaluation	Striaghtforward/Limited	Complex/Extended
Run time system	Limited	Extended

6- CPU time (A) = CC(A) * CT (A) = CC(A) / f(A)
CPU time (A) = $50 \times 500 \times 10^6 = 25 \times 10^9$ Cycles
CPU time (B) = CC(B) * CT (B) = CC(B) / f(B)

$$\text{Clock rate (B)} = \frac{2.5 \times 25 \times 10^9}{20} = 3.125 \text{ GHz}$$

7- Assume that the same number of instructions in both cases =N

$$CPU \ \text{Clock Cycle (A)} = \textit{Instruction count} \times CPI = 4 \times N$$

$$CPU \ \text{Clock Cycle (B)} = \textit{Instruction count} \times CPI = 2.5 \times N$$

$$CPU \ \textit{time} \ (A) = \text{CPU clock cycle (A)} \times \textit{Clock cycle time} \ \text{(A)}$$

$$Clock\ cycle\ time\ (A) = \frac{50\times10^{-9}}{4\times N} = \frac{1.25\times10^{-9}}{N}\ \text{sec}$$

$$Clock\ cycle\ time\ (B) = \frac{65\times10^{-9}}{2.5\times N} = \frac{26\times10^{-9}}{N}\ \text{sec}$$

For any N, clock cycle time (A) < Clock cycle time (B), i.e., Machine A is faster than Machine B.

8-

$$CPI_1 = \frac{CPU\ Clock\ Cycles}{Instruction\ Count} = \frac{(2\times1+1\times3+2\times4)\times10^6}{(2+1+2)\times10^6} = 2.6$$

$$CPI_2 = \frac{CPU\ Clock\ Cycles}{Instruction\ Count} = \frac{(4\times1+3\times3+1\times4)\times10^6}{(4+3+1)\times10^6} = 2.125$$

$$CPU\ Time_1 = \frac{Instruction\ Count\times CPI_1}{Clock\ Rate} = \frac{13}{Clock\ Rate}\ \text{sec}$$

$$CPU\ Time_2 = \frac{Instruction\ Count\times CPI_2}{Clock\ Rate} = \frac{17}{Clock\ Rate}\ \text{sec}$$

For the same clock rate, CPU Time1 < CPU Time 2, i.e., code sequence 1 is faster than code sequence 2.

9-

$$CPI_1 = \frac{(15\times2+5\times5+3\times7)\times10^6}{23\times10^6} = 3.3$$

$$CPI_2 = \frac{(25\times2+2\times5+2\times7)\times10^6}{29\times10^6} = 2.55$$

$$MIPS_1 = \frac{Clock\ Rate_1}{CPI_1} = \frac{500\times10^6}{3.3\times10^6} = 151.515$$

$$MIPS_2 = \frac{Clock\ Rate_2}{CPI_2} = \frac{500\times10^6}{2.55\times10^6} = 196.1$$

$$\text{CPU time} = \frac{\text{Instruction count} * \text{CPI}}{\text{Clock rate}}$$

$$CPU\ Time_1 = \frac{23\times10^6\times3.3}{500\times10^6} = 152\ \text{m.sec.}$$

$$CPU\ Time_1 = \frac{23\times10^6\times3.3}{500\times10^6} = 152\ \text{m.sec.}$$

10- One Enhancement:

$$(a)\ SU_a = \frac{1}{(1-F_a)+\frac{F_a}{SU_1}} = \frac{30}{(1-0.25)+\frac{0.25}{30}} = 1.3186$$

(b) $SU_b = \dfrac{1}{(1-F_b)+\dfrac{F_b}{SU_2}} = \dfrac{20}{14.3} = 1.3986$

(c) $SU_c = \dfrac{1}{(1-F_c)+\dfrac{F_c}{SU_3}} = \dfrac{15}{8.7} = 1.7241$

Two Enhancements:

(a) a & b:

$$SU_{a-b} = \frac{1}{(1-F_a-F_b)+\dfrac{F_a}{SU_1}+\dfrac{F_b}{SU_2}} = \frac{1}{(1-0.25-0.3)+\dfrac{0.25}{30}+\dfrac{0.3}{20}} = 2.11$$

(b) a & c

$$SU_{a-c} = \frac{1}{(1-F_a-F_c)+\dfrac{F_a}{SU_1}+\dfrac{F_c}{SU_3}} = \frac{1}{(1-0.25-0.45)+\dfrac{0.25}{30}+\dfrac{0.45}{15}} = 2.955$$

(c) b & c

$$SU_{b-c} = \frac{1}{(1-F_b-F_c)+\dfrac{F_b}{SU_2}+\dfrac{F_c}{SU_3}} = \frac{1}{(1-0.3-0.45)+\dfrac{0.3}{20}+\dfrac{0.45}{15}} = 3.389$$

Chapter (2)

1. Write a program using the addressing modes and the instruction types presented in Sections 2.2 and 2.3 to reverse the bits stored in a 16-bit register R_0.

2. Consider a computer that has a number of registers such that the three registers $R_0 = 1500, R_1 = 4500$ and $R_2 = 1000$. Show the effective address of memory and the registers contents in each of the following instructions (assume that all numbers are decimal).

   (a) $ADD \quad (R_0)+, R_2$
   (b) $SUTRACT \quad -(R_1), R_2$
   (c) $MOVE \quad 500(R_0), R_2$
   (d) $LOAD \quad \#5000, R_2$
   (e) $STORE \quad R_0, 100(R_2)$

3. Assume that the top of the stack in a program is pointed to by the register *SP*. You are required to write program segments to perform each of the following tasks (assume that only the following addressing modes available are: indexed, autoincrement, and autodecrement).

   (a) Pop the top three elements of the stack, add them, and push the result back onto the stack.
   (b) Pop the top two elements of the stack, subtract them, and push the results back onto the stack.
   (c) Push five elements (one at a time) onto the stack.
   (d) Remove the top five elements from the top of the stack.
   (e) Copy the third element from the top of the stack into register R_0.

4. You are required to write a program segment that can perform the operation $C \leftarrow A + B$ where each of A and B represents a set of 100 memory locations each storing a value such that the set of values represented by A are stored starting at memory location 1000 and those represented by B are stored starting at memory location 2000. The results should be stored starting at memory location 3000. The above operation is to be performed using each of the following instruction classes.

   (a) A machine with one-address instructions.
   (b) A machine with one-and-half instructions.
   (c) A machine with two-address instructions.
   (d) A machine with three-address instructions.
   (e) A machine with zero-address instructions.

5. Write program segments that perform the operation $C \leftarrow C + A \times B$ using each of the instruction classes indicated in exercise 4 above. Assume that A, B, and C are memory addresses.

6. Assume that a series of five tests has been offered to a class consisting of 50 students. The score obtained by students in each of the five tests are stored sequentially in memory locations starting respectively at memory locations 1000, 2000, 3000, 4000, and 5000. You are required to write a program that calculates the average score obtained by each student in the five tests and store the same in memory locations starting at memory location 6000. Each student is identified by his/her student ID. You may assume that students' IDs are sequential.

7. Repeat exercise 6 above assuming that the memory used is byte addressable while each score occupies 32-bit.

8. Rewrite the same program as in exercise 6 above assuming that the students' IDs are not sequential, i.e., each student ID is to be used as a pointer to his/her test scores.

9. Repeat exercise 6 above assuming that the students scores are stored in an array S(50,5), i.e., each row holds the scores obtained by a student (each score in a column of the same row) and that the first element of the array, i.e., S (0,0) is stored in memory location 4000. The scores are stored row-wise, i.e., one row after the other. The average score obtained by each student is to be stored at a memory location pointed to by his/her ID.

10. Repeat exercise 9 above assuming that your job is to write a subroutine to perform the same task as in exercise 9. Assume that the number of students, the number of tests, and the location of the first element in the array are to be passed to the subroutine as parameters in registers $R_1, R_2,$ and R_3, respectively.

Chapter (3)

1. What is the difference between each of the following pairs:
   - *Compilers and assemblers*
   - *Source code and target code*
   - *Mnemonics and Hexadecimal representation*
   - *Pseudo instructions and instructions*
   - *Labels and addresses*
   - *Symbol table and opcode table*
   - *Program Counter (PC) and Instruction Location Counter (ILC)*

2. Using the assembly language of the simple processor in Section 3.1, write assembly code segments to do the following operation:
   - Swap two numbers
   - Logical OR
   - Negation

3. Add input/output instructions to the instruction set of the simple processor in Section 3.1 and write an assembly program to find the Fibonacci sequence.

4. Obtain the machine language code of the multiplication assembly program given in Section 3.2.

5. With the great advances in high level languages and compilers, some people argue that assembly language is not important anymore. Give some arguments for and against this view.

6. Write a program segment using the instruction of the *X86 family* to compute the $\sum_{i=1}^{200} X_i Y_i$, where X_i and Y_i are signed 8-bit numbers. Assume that no overflow will occur.

7. Write a subroutine using the *X86* instructions that can be called by a main program in a different code segment. The subroutine will multiply a signed 16-bit number in CX by a signed 8-bit number in AL. The main program will call this subroutine, store the result in two consecutive memory words, and stop. Assume that SI and DI contain the signed 8-bit and the 16-bit numbers, respectively.

8. Write a program using the *X86* instructions to compare a source string of 100 words pointed to by an offset of 2000H in DS with a destination string pointed to by an offset 4000H in DS.

9. Write a program using the *X86* instructions to generate the first 10 numbers of the Fibonacci series, i.e. to generate the series 1, 1, 2, 3, 5, 8, 13, 21, 34.

10. Write a program using the *X86* instructions to convert a word of text from uppercase to lowercase. Assume that the word consists of ASCII characters stored in successive memory locations starting at location *START* and ending at location *FINISH*.

Chapter (4)

1.

	Sign-and-magnitude	2's complement
26	0000011010	0000011010
-123	1001111011	1110000101

2. Compute the decimal value of the binary number 1011 1101 0101 0110 if the given number represents unsigned integer. Repeat if the number represents 2's complement. Repeat if the number represents sign-magnitude integer.

	Unsigned	2's complement	Sign-magnitude
1011 1101 0101 0110	47970	17066	-15202

3.

010110	011001	110111	100001	111111	011010
+ 001001	+010000	+101011	-011101	-000111	-100010
011111	101001	100010	000100	111000	111000
No O.F.	O.F.	No O.F.	O.F.	O.F.	O.F.

4.

M	A	Q	Q(-1)	Operation	Remarks
010111	000000	110110	0		
010111	000000	011011	0	ASR	First Cycle
010111	101001	011011	0	A← A-M	
010111	110100	101101	1	ASR	Second Cycle
010111	111010	010110	1	ASR	Third Cycle
010111	010001	010110	1	A←A+M	
010111	001000	101011	0	ASR	Forth Cycle
010111	110001	101011	0	A←A-M	
010111	111000	1101011	1	ASR	Fifth Cycle
010111	**111100**	**011010**	1	ASR	Sixth Cycle

M=010111	M=110011	M=110101	M=1111
Q=110110	Q=101100	Q=011011	Q=1111
11100011010=-230	000100000100=260	111011010111=-297	00000001=1

5. Divide each of the following pairs of signed 2's complement numbers using both the restoring and the nonrestoring algorithms.

X=010111	X=110011	X=110101	X=1111
D=110110	D=101100	D=011011	D=1111

6.

A	$+0.1011011\times 2^{1}$
B	-0.1101010×2^{-1}
Align B	-0.001101010×2^{1}
A+B	0 10001 000001
A-B	0 10001 110101
A×B	1 10000 001001 011010 1110

7.

$$c_0 = g_0$$
$$c_1 = g_1 + P_1 c_0 = g_1 + P_1(g_0) = g_1 + g_0 P_1$$
$$c_2 = g_2 + P_2 c_1 = g_2 + P_2(g_1 + G_0 P_1) = g_2 + g_0 P_1 P_2 + g_1 P_2$$
$$c_3 = g_3 + P_3 c_2 = g_3 + P_3(g_2 + g_1 P_2 + g_0 P_1 P_2) = g_3 + g_2 P_3 + g_1 P_2 P_3 + g_0 P_1 P_2 P_3$$

•

8. Design a BCD adder using a 4-bit binary adder and the least number of logic gates. The adder should receive two 4-bit numbers A and B and should produce 4-bit sum and a carry output.

9. Show a design of a 16-bit CLA that uses the 4-bit CLA block shown in Fig. 4.6. Compute the delay and the area (in terms of the number of logic gates required).

10. Compare the longest path delay from input to output of a 32-bit adder using 4-bit CLA adder blocks in a multi-level architecture with that of a 32-bit CRT adder. Assume that a gate delay is given by T_g.

11. Convert each of the following decimal numbers to their IEEE single-precision floating-point counterparts.

(a) -76
(b) 0.92
(c) 5.3125
(d) -0.000072
(e) 8.04×10^{21}

12. Convert of the following IEEE single-precision floating-point numbers to their decimal counterparts.

(a) 6589 00000
(b) 807B 00000H
(c) CDEF 0000H

13. Complete the logic design of the array multiplier shown in Fig. 4.7.

14. Complete the design of the control logic shown in Fig. 4.8.

15. Provide a complete logic design for the Control Logic indicated in Fig. 4.11.

Chapter (5)

1. How many instruction bits are required to specify the following:
a) two operand registers and one result register in a machine that has 64 general-purpose registers?
b) three memory addresses in machine with 64KB of main memory?

2. Show the micro-operations of the *load, store,* and *jump* instructions using:
- One-bus system
- Two-bus system
- Three-bus system

3. Add control signals to all the tables in Section 5.4

4. Data movement within the CPU can be performed using several different ways. Contrast the following methods in terms of their advantages and disadvantages:

a) Dedicated connections
b) One-bus datapath
c) Two-bus datapath
d) Three-bus datapath

5. Find a method of encoding the microinstructions described by the following table so that the minimum number of control bits is used and all inherent parallelism among the microoperations is preserved.

Microinstruction	**Control signals activated**
I1	*a, b, c, d, e*
I2	*a, d, f, g*
I3	*b, h*
I4	*c*
I5	*c, e, g, i*
I6	*a, h, j*

6. Suppose that the instruction set of a machine has three instructions: Inst-1, Inst-2, and Inst-3; and A, B, C, D, E, F, G and H are the control lines. The following table shows the control lines that should be activated for the three instructions at the three steps T0, T1, and T2.

Step	Inst-1	Inst-2	Inst-3
T0	D, B, E	F, H, G	E, H
T1	C, A, H	G	D, A, C
T2	G, C	B, C	

a) Hardwired approach
i) Write Boolean expressions for all the control lines A- G.
ii) Draw the logic circuit for each control line

b) Microprogramming approach
i) Assuming a horizontal representation, write down the microprogram for instructions Inst-1. Indicate the microinstruction size.
ii) If we allow both horizontal and vertical representation, what would be the best grouping? What is the microinstruction size? Write the microprogram of Inst-1.

7. A certain processor has a microinstruction format containing 10 separate control fields $C_0:C_9$. Each C_i can activate any one of n_i distinct control lines, where n_i is specified as follows:

i:	0	1	2	3	4	5	6	7	8	9
n_i:	4	4	3	11	9	16	7	1	8	22

a) What is the minimum number of control bits needed to represent the 10 control fields?
b) What is the maximum number of control bits needed if a purely horizontal format is used for all control information?

8. What are the main differences between the following pairs:
a) Vertical and horizontal microinstructions
b) Microprogramming and hardwired control

9. Using the single-bus architecture, generate the necessary control signals, in the proper order (with minimum number of micro-instructions), for *Conditional Branch* instruction.

10. Write a micro-program for the fetch instruction using the one-bus datapath and the two-bus datapath.

Chapter (6)

1. $t_a = t_c + \frac{t_m}{n} < 60\,n\text{sec}$, n is the interleaving factor. $20 + \frac{100}{n} < 60\,n\text{sec}$, n > 2.5, i.e., n=4 (n is usually a power of 2). $t_a = 20 + \frac{100}{4} = 45\,n\text{sec}$

2.
$$t_a = h_c t_c + (1-h_c)[h_m t_m + (1-h_m)t_d] = 0.9\times 20 + 0.1\times[0.95\times 100 + 0.05\times 10^6] = 5027.5\,n\text{sec}$$

3. Block size B = 1 word = 32 bits =4 bytes, M = number of blocks in MM = $\frac{16M}{4} = 2^{22}\ blocks$. N = number of blocks in cache= $\frac{8K}{4} = 2^{11}\ blocks$.

(a) Direct mapping with block size of one word.
Word field = log B = 2 bits. Block field = log N = 11 bits. Tag field = log (M/N)=11 bits.

(b) Direct mapping with a block size of eight words.
Word field = log B = 5 bits. $N = \frac{8K}{32} = 2^8$. Block field = log N = 8 bits. $M = \frac{16M}{32} = 2^{19}$.
Tag field = log (M/N) = 11 bits.

(c) Associative mapping with a block size of eight words.
Word field = log 32 = 5 bits. $M = \frac{2^{24}}{32} = 2^{19}$. Tag field = log M = 19 bits.

(d) Set-associative mapping with a set size of four block and a block size of one word.
Word field = log 2^2 =2 bits. $N = \frac{8K}{4} = 2^{11}$. Set size = 4×4=16 bytes. S = number of sets in cache=$N = \frac{N}{16} = 2^7$. Set Field = log S= 7 bits. Tag field = log (M/S)= 15 bits.

4.
Assume column-major order. The elements of the array are stored in the main memory as follows:

M_0 M_1 M_0

6 4 4 4 47 4 4 4 48 6 4 4 4 47 4 4 4 48 6 4 4 4 47 4 4 4 48

A00	A10	A20	A30	A40	A50	A60	A70	A01	A11	A21	A31	A41	A51	A61	A71	..	..	..	..	..	..	..	..	A07	A17	A27	A37	A47	A57	A67	A77

Sequence of access is as follows (row wise):

A00	A01	A02	A03	A04	A05	A06	A07
A10	A11	A12	A13	A14	A15	A16	A17
A20	A21	A22	A23	A24	A25	A26	A27
A30	A31	A32	A33	A34	A35	A36	A37
A40	A41	A42	A43	A44	A45	A46	A47
A50	A51	A52	A53	A54	A55	A56	A57
A60	A61	A62	A63	A64	A65	A66	A67
A70	A71	A72	A73	A74	A75	A76	A77

A00 $\rightarrow$ cache miss $\rightarrow$ Access time =110 nsec
A01 $\rightarrow$ cache hit $\rightarrow$ Access time = 10 nsec
.
.

A07 $\rightarrow$ cache hit $\rightarrow$ Access time = 10 nsec
Total Access time = 180 nsec.
Total Access time for 8 rows = 1440 nsec
Average Access time per element = 1440/64=22.5 nsec

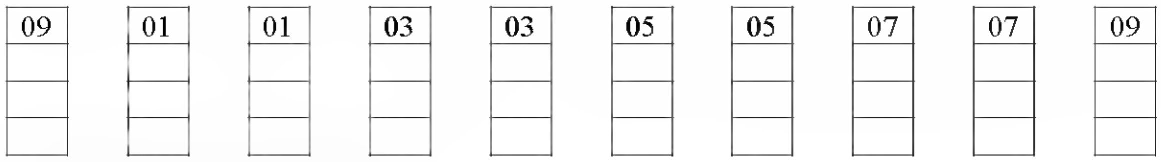
5. The elements of the array are stored in the main memory as follows:

A00	A10	A20	A30	A01	A11	A21	A31	A02	A12	A22	A32	A03	A13	A23	A33	..	..	..	..	..	..	..	..					A09	A19	A29	A39

(a) Direct Mapping:

j=0	j=1	j=2	j=3	j=4	j=5	j=6	j=7	j=8	j=9
A 00	A 00	A 02	A 02	A 04	A 04	A 06	A 06	A 08	A 08
	A 01	A 01	A 03	A 03	A 05	A 05	A 07	A 07	A 09

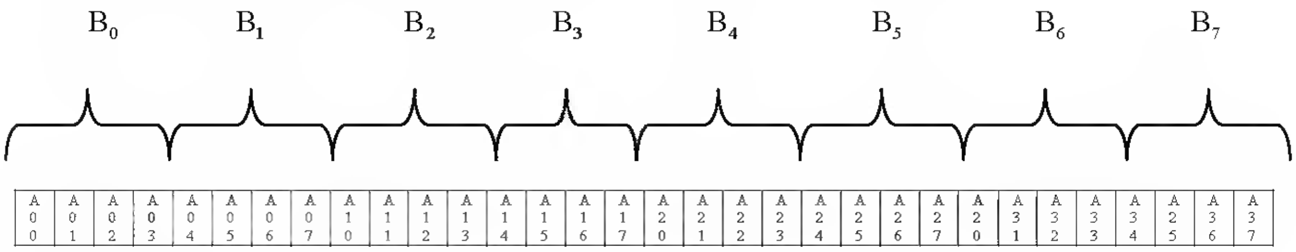
k=0	k=1	k=2	k=3	k=4	k=5	k=6	k=7	k=8	k=9
A 00	A 00	A 02	A 02	A 04	A 04	A 06	A 06	A 08	A 08
A	A	A	A	A	A	A	A	A	A



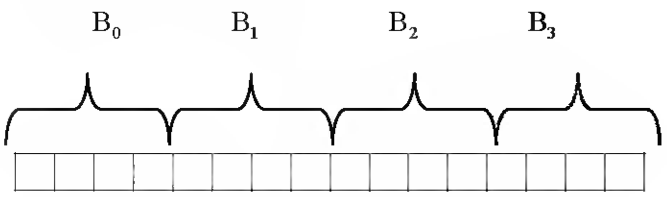
Number of Replacements = 18
Cache Utilization = 2/8 =25%

(b) Associate Mapping
Use similar trace to show that the number of Replacements = 12 and Cache Utilization = 100%
(c) Set Associate Mapping
Use similar trace that the number of Replacements = 16 and Cache Utilization 50%.

6. Elements of the array in the Main Memory



Cache Memory (N=4)



(a) Direct Mapping (j←i mode 4)

$a_{0,0} \rightarrow miss \rightarrow B_0(Cache) \leftarrow B_0(MM)$

$a_{0,1}, \rightarrow hit$

$a_{0,2} \rightarrow hit$

$a_{0,3} \rightarrow hit$

$a_{0,4}, \rightarrow miss \rightarrow B_1(Cache) \leftarrow B_1(MM)$

$a_{0,5} \rightarrow hit$

$a_{0,6} \rightarrow hit$

$a_{0,7} \rightarrow hit$

$a_{1,0} \rightarrow miss \rightarrow B_2(cache) \leftarrow B_2(MM)$

$a_{1,1} \rightarrow hit$

$a_{1,2} \rightarrow hit$

$a_{1,3} \rightarrow hit$

$a_{1,4} \rightarrow miss \rightarrow B_3(cache) \leftarrow B_3(MM)$

$a_{1,5} \rightarrow hit$

$a_{1,6} \rightarrow hit$

$a_{1,7} \rightarrow hit$

$a_{2,0} \rightarrow miss \rightarrow B_0(cache) \leftarrow B_4(MM) \rightarrow \text{Re}\,placement$

$a_{2,1} \rightarrow hit$

$a_{2,2} \rightarrow hit$

$a_{2,3} \rightarrow hit$

$a_{2,4} \rightarrow miss \rightarrow B_1(cache) \leftarrow B_5(MM) \rightarrow \text{Re}\,placement$

$a_{2,5} \rightarrow hit$

$a_{2,6} \rightarrow hit$

$a_{2,7} \rightarrow hit$

$a_{3,0} \rightarrow miss \rightarrow B_2(cache) \leftarrow B_6(MM) \rightarrow \text{Re}\,placement$

$a_{3,1}, \rightarrow hit$

$a_{3,2} \rightarrow hit$

$a_{3,3} \rightarrow hit$

$a_{3,4} \rightarrow miss \rightarrow B_3(cache) \leftarrow B_7(MM) \rightarrow \text{Re}\,placement$

$a_{3,5} \rightarrow hit$

$a_{3,6} \rightarrow hit$

$a_{3,7} \rightarrow hit$

Number of Replacements= 4
Cache Utilization = 100%

(b) Associative Mapping

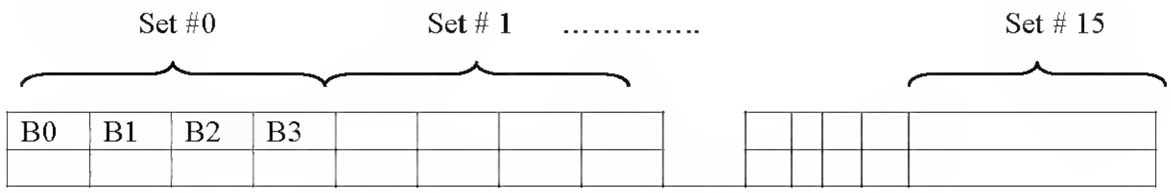
A similar trace can be made to show that Number of Replacements = 4 and Cache utilization = 100%

(c) Set Associative Mapping ($s \leftarrow i \bmod S$) and S = 2.
A similar trace can be made to show that Number of Replacements = 6 and Cache utilization = 50%

7. Set Associative Mapping with

$$\left.\begin{array}{l}\text{Word Field} = \log 64 = 6 \text{ bits}\\ \text{Set Field} = \log 16 = 4 \text{ bits}\\ \text{Tag Field} = \log 1\text{K} = 10 \text{ bits}\end{array}\right\} \quad \text{sum} = 20 \text{ bits} = \log 1\text{M}$$

LRU:



Cache Set Number	MM Blocks competing for cache set
0	0, 16, 32, 48, 64
1	1, 17, 33, 49, 65
2	2, 18, 34, 50, 66
3	3, 19, 35, 51, 67

First Pass: 64 MM blocks are brought into the cache.
On each of the next Nine passes, 48 MM blocks are to be found in the cache, while the remaining 20 MM blocks are to be fetches from MM.
Time with no cache = $10\times10\tau\times68=6800\ \tau$ units
Time with cache = $1\times11\ \tau\times68 + 9(48\times1\ \tau+20\times11\tau)=3160\ \tau$ units
Improvement factor = $6800\ \tau/3160\ \tau=2.15$

MRU:

$$\text{Cache set \#0}\left\{\begin{array}{l}\text{First Pass: MM blocks 0, 1, 2, ……., 63 will fill the cache}\\ \text{Second Pass: MM block 48 will replace MM block 32}\\ \text{Third Pass: MM block 32 will replace MM 16}\\ \text{Forth Pass: MM block 16 will replace MM 0}\\ \text{Fifth Pass: MM block 0 will replace MM 64 and MM 64 will replace MM 48}\\ \text{….}\\ \text{…..}\end{array}\right.$$

In passes 2 to 10, a total of 11 replacements. Similar situations occur in sets 1, 2, and 3. Sets 4 to 15 will have no contention.
Time with cache = $1\times11\ \tau\times68 + 4\times\ 11\times11+1\ \tau\times(9\times68\text{-}44) = 18000\ \tau$ units
Improvement factor = $6800\ \tau/1800\ \tau=3.7$

Chapter (7)

1. Five Access combinations are observed

(a) VA → TLB (hit) → PA → Cache (hit)
(b) VA → TLB (hit) → PA → Cache (miss) → update cache
(c) VA → TLB (miss) → PT (hit) → PA → update TLB → cache hit
(d) VA → TLB (miss) → PT (hit) → PA → update TLB → cache miss → update cache
(e) VA → TLB (miss) → PT (miss) → HD → update all

(a) Access probability = 0.9×0.95= 0.855
(b) Access probability= 0.9×0.05×0.8=0.036
(c) Access probability= 0.1×0.8×0.95=0.076
(d) Access probability= 0.1×0.8×0.05×0.8= 0.0032
(e) Access probability= 0.1×0.2=0.02

(a) Access time= 0.9×25+0.95×25=46.25 nsec
(b) Access time= 0.9×25+0.05×25+0.8×250= 224.7 nsec
(c) Access time= 0.1×25+0.8×250+0.95*25= 226.25 nsec
(d) Access time= 0.1×25+0.8*250+25+25*0.05+25=253.75 nsec
(e) Access time= 0.1*25+0.2*250+100,000,000+250+25+25=0.1000003525 sec

2.
(a)

```
For I = 0 → 63
    Max ← a(I,0)
    For J = 0 → 63
        If a(I,J) > max then max ← a(I,J)
    End For

    For J = 0 → 63
        A(I,J) ← a(I,J)/max
    End For
End For
```

(b) I = 0

J = 0	1	2	3	………..	63
A00	a01	a02	a03		a0,63
PF	PF	PF	PF		PF

J = 0	1	2	3	……….	63
A00	a01	a02	a03		a0,63
PF	PF	PF	PF		PF

A total of 128 PFs in each row search.
Total number of PFs = 128 × 64 = 8192 PFs

(c) I = 0

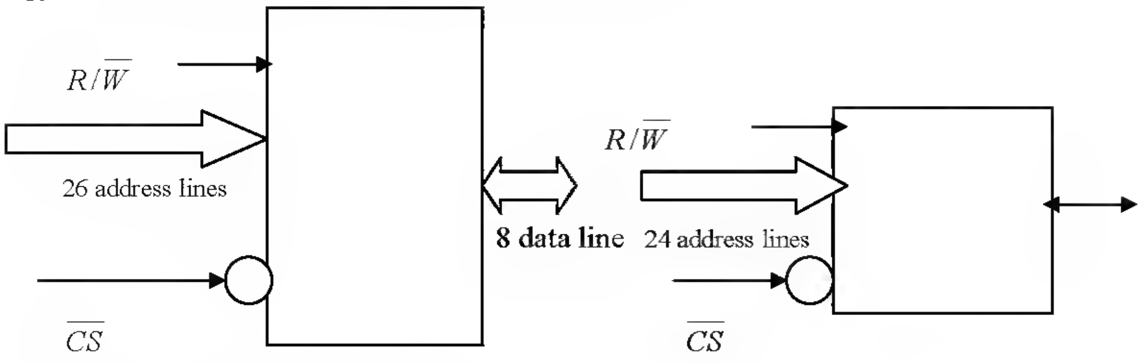
J = 0	1	2	3	……..	63
A00	a01	a02	a03		a0,63
PH	H	H	H		H

J = 0	1	2	3	……..	63
A00	a01	a02	a03		a0,63
H	H	H	H		H

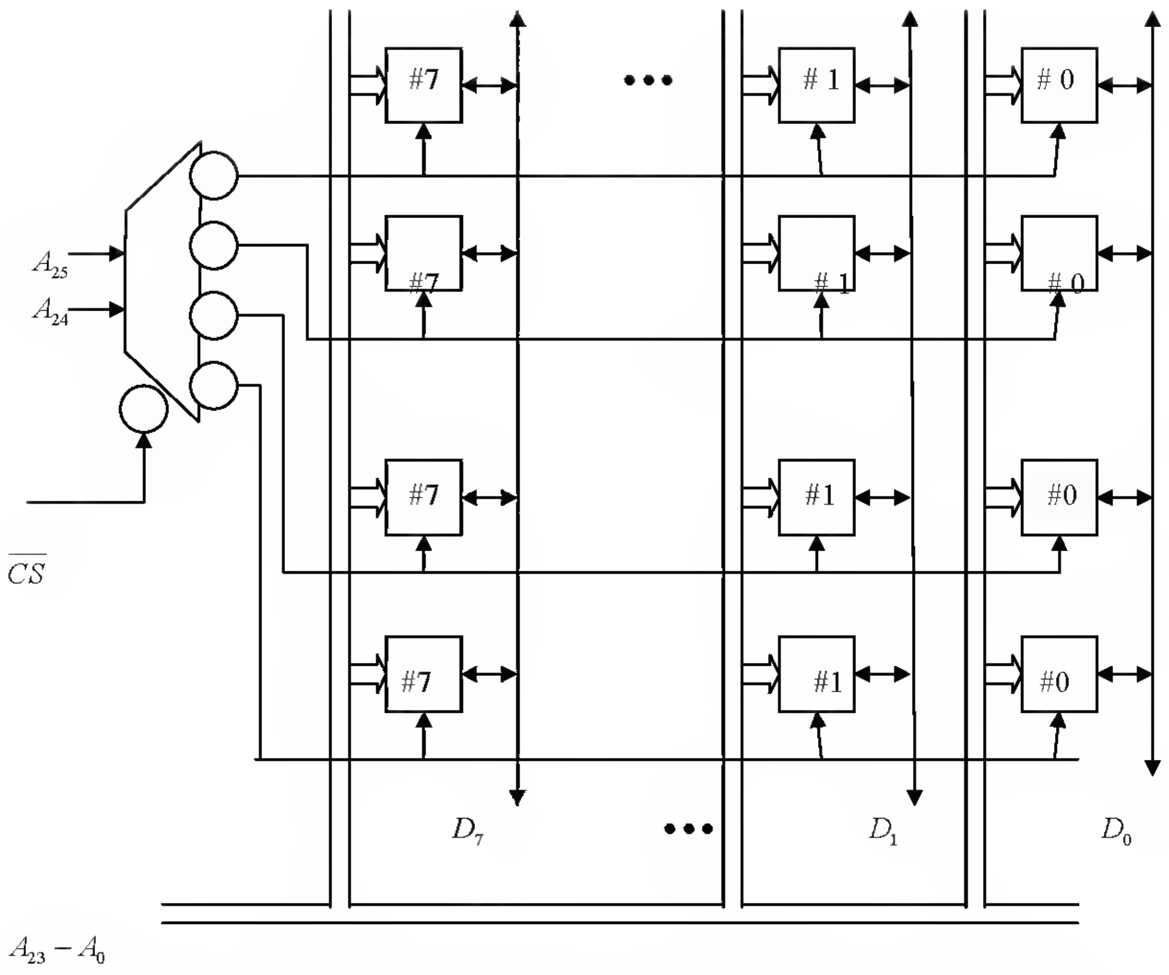
One PF in each row search.
Total number of PFs = 64 PFs

(d) Case 1: Estimated Time = 8192× 100 = 81.92 sec.
Case 2: Estimated Time = 64× 100 = 6.4 sec.

8.



$$Number\ of\ chips = \frac{64 \times 2^{20} \times 8}{16 \times 2^{20} \times 1} = 32\ chips$$



9. Consider the following stream of page requests: 1,2,3,4,5,1,2,3,4,5,1,2,3,4,5. Assume that the main memory consists of FOUR page frame. Show a trace of the status of the page frames in the MM and estimate the hit ratio assuming each of the following page replacement algorithms.

(a) FIFO

1	2	3	4	5	1	2	3	4	5	1	2	3	4	5	
	1	1	1	1	5	5	5	5	4	4	4	4	3	3	3
		2	2	2	2	1	1	1	1	5	5	5	5	4	4
			3	3	3	3	3	2	2	2	1	1	1	1	5
				4	4	4	4	3	3	3	3	2	2	2	2
	PF	PF	PF	PF	PF	PF	PF	PF	PF	PF	PF	PF	PF	PF	PF

Hit Ratio = 0%

(b) LRU

Same as above

(c) FI-NU-FO
Same as above

10.
I = 1

J =1	2	3	4	5	6
A11	A12	A13	A14	A15	A16
F	H	F	H	F	H

Number of PFs in each loop = 20/2 = 10
Total number of PFs = 20×10= 200 PFs.

```
for I  = 1 to 20 do
for J  = 1 to  20 do
A[I, J] = 0;
```

11. See the Examples given in the Chapter. Follow these examples in analyzing your picked up system.

Chapter (8)

1. Conduct an Internet search on I/O devices and prepare a table categorizing the different devices into separate categories, for example input, output, character based block based, etc. For every entry in the table, indicate its speed, interface, and category.

2. What are the advantages and disadvantages of isolated versus memory mapped I/O.

3. Show how data transfer from Disk to memory is conducted under each of the following I/O schemes: Programmed I/O, interrupt driven I/O, and DMA. Show the steps taken in each case.

4. If an interrupt requires 50m s of overhead time, and poling requires 5m s per device, describe different situations where each seems better than the other.

5. What entities in a computer system does a device driver communicate with? What are the functions of a device driver? List all operations.

6. What types of operations is DMA used to accelerate?

7. A DMA module is transferring data to memory using cycle stealing from a device that transmits data at rate 19200 bits per second. The speed of the CPU is 3 MIPS. By how much would the DMA module affect the performance of the CPU.

8. Describe the scenarios in which a synchronous bus would outperform an asynchronous bus and vice versa.

9. Discuss the advantages and disadvantages of the different bus arbitration policies covered in the chapter. Prepare a contract table that compares the arbitration techniques from both the implementation and operational aspects.

Chapter (9)

1.

(a) Speed up S(n) = $= \dfrac{m \times n}{n+m-1+p \times m(n-1)} = \dfrac{500 \times 5}{5+499+0.3 \times 500(5-1)} = 2.26$

(b) S(n)

$\dfrac{2500}{504+2000p} \geq 4$, $p \leq 0.0605$, number of branch instructions $\leq 0.0605 \times 500 \leq 32.5$

(c) $\dfrac{2500}{504+2000p} \geq 5$, it is impossible to have a speed up of at least 5

2.

Case #1: # instructions per cycle = 1000/1300= 0.769

Case #2: Number of cycles = 1300-255 = 1045

# instructions per cycle = 1000/1045 = 0.9569

percentage gain = (0.9569-0.769)/0.769%= 24.4%

3.

(a)

			1	2	3				4	5	6	7	8	9				10	11	12	13	14	15				16	17	18	19	20
		1	2	3				4	5	6	7	8	9				10	11	12	13	14	15				16	17	18	19	20	
	1	2	3				4	5	6	7	8	9				10	11	12	13	14	15				16	17	18	19	20		
1	2	3				4	5	6	7	8	9				10	11	12	13	14	15				16	17	18	19	20			

Number of time slots = 35

(1) speed up = 20×4/35 = 2.29

(2) throughput = 20/35= 0.57

(3) efficiency = 20/35 = 0.57

(4) number of cycles per instruction 35/20 = 1.75

(b) Number of time slots = 42

(1) speed up = 80/42 = 1.9

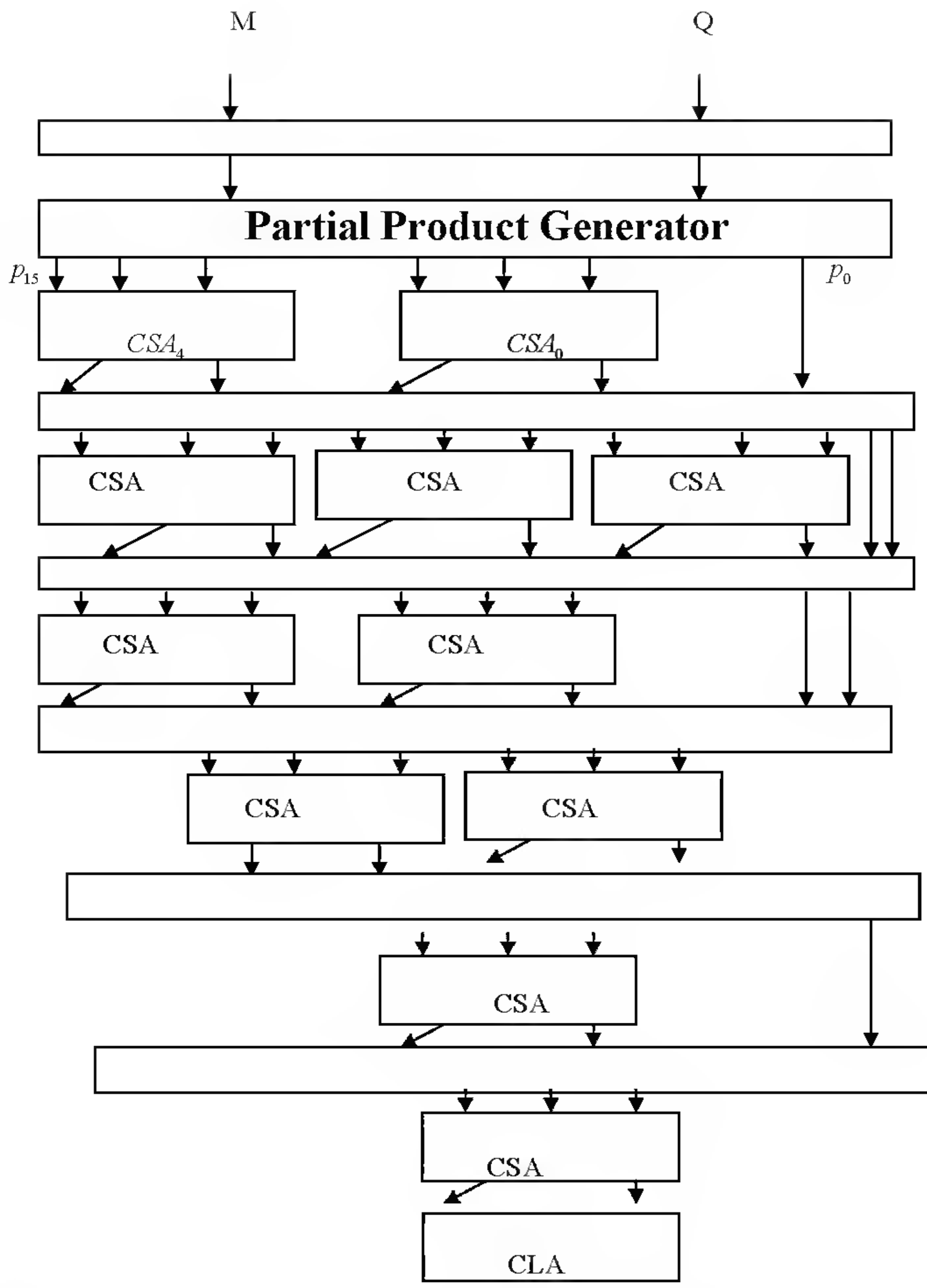
(2) throughput = 20/42 = 0.476

(3) efficiency = 20/42 = 0.476

(4) number of cycles per instruction = 42/20 = 2.1

4.

$$P = \sum_{i=0}^{15} P_i = M \times Q = M \times (Q_{15} \times 2^{15} + \dots\dots + Q_0 \times 2^0) = \sum_{i=0}^{15} M \times Q_i \times 2^i$$



5.

		1	2		3	4		5	6		7	8		9	10		11	12		13	14		15	16		17	18		19	20
	1	2		3	4		5	6		7	8		9	10		11	12		13	14		15	16		17	18		19	20	
1	2		3	4		5	6		7	8		9	10		11	12		13	14		15	16		17	18		19	20		

Number of time slots = 31

6.

Number of time slots = m+n-1 + Number of branch instructions × (n-1)

= 36+5-1+5×(5-1)=60

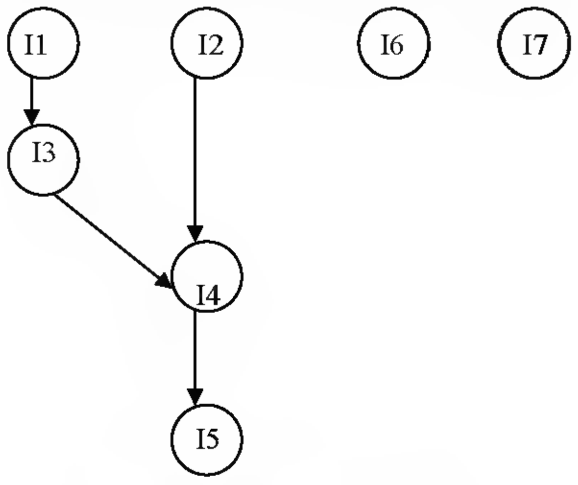
Average processing time = 60/36=1.67

Student should show the space-time diagram for the processing of these 36 instructions.

7. A computer has a 5-stage instruction pipeline of one cycle each. The five stages are: Instruction Fetch (IF), Instruction Decode (ID), Operand Fetch (OF), Instruction Execution (IE), and Operand Store (OS). Consider the following code sequence, which is to be run on this computer.

```
            I1    Load  -1, R1;       R1 ← -1;
            I2    Load  5, R2;        R2 ← 5;
Again:      I3    Sub   R2, 1, R2     R2 ← R2 - 1;
            I4    Add   R1, R2, R3;   R3 ← R1 + R2;
            I5    Bnn   Again;        branch to Again if result is Not
               Negative;
            I6    Add   R4, R5, R6;   R6 ← R4 + R5;
            I7    Add   R6, R4, R7;   R7 ← R4 + R6;
```

(a) Analyze the execution of the above piece of code in order to calculate the number of cycles needed to execute the above code without pipelining, assuming that each instruction requires exactly 5 cycles to execute.



Number of cycles = 5×(2+3×5+2)=95 cycles

(b) Construct the space-time chart to show that the number of cycles = 41 cycles.

(c) Construct the space-time chart taking data dependency into consideration to show that the number of cycles = 54 cycles.

(d) Percentage saving (a) = (95-41)/41 % = 131.7 %

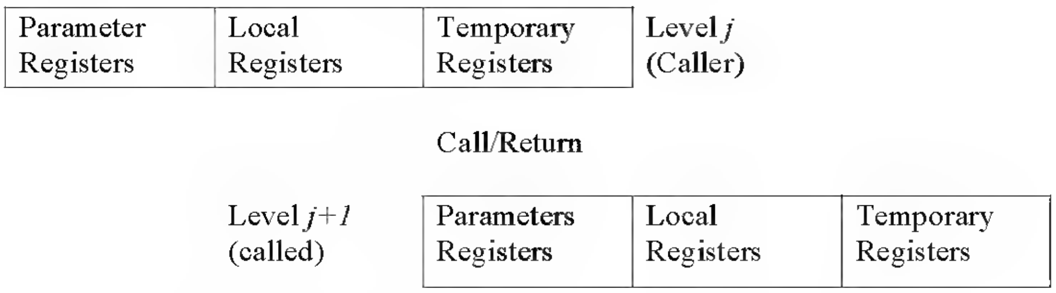
Percentage saving (b) = (95-54)/54 % = 75.9%

Chapter (10)

1. Main principles are
   (1) One instruction per machine cycle
   (2) Fixed instruction length
   (3) Reduced/Simplified addressing modes
   (4) Use of Register Operations except for Load/Store instructions
   (5) Simplified/No Complex instructions

2. The hardware approach depends on the availability of large number of registers, e.g., 1K to 4K registers in addition to the use of the register window concept (see Problem 3 below) in minimizing the use of memory operations. This approach is adopted in the Berkeley RISC machines. The software approach depends on the use of a smart compiler in rearranging instructions and/or allocating registers to the most frequently used variables in minimizing the use of memory operations. This approach is adopted in the Stanford MIPS.

3. Register Window:
(a) Multiple small set of registers, each is assigned to a different procedure
(b) Procedure call automatically switches the register window in use to a new one
(c) Only one window is visible at a given time
(d) Set of fixed number of registers are available to all procedures as global registers in order to hold global variables
Window Overlapping:



Examples: Berkeley RISC: 8 windows, 16 registers each.
Pyramids RSIC: 16 windows, 32 registers each.

4. Students are required to prepare a report on a recent RISC machine. Use information given in the chapter and also make use of available information in the Internet.

5. Students are required to prepare a report on a recent CISC machine. Use information given in the chapter and also make use of available information in the Internet.

6. Students are required to prepare a report as advocates for the RISC approach.

7. Students are required to prepare a report as advocates for the CISC philosophy.

Chapter (11)

1. Consider the five classifications of computer architectures discussed in this chapter. You are required to provide a list showing the advantages and disadvantages of each classification in view of the degree in which each classification satisfies the purpose for which a classification is needed.

2. You are required to derive, out of the five provided classifications, a new classification that outperforms each of the five classifications. Provide, in a tabular form, the additional advantages and eliminated shortcomings of the proposed classification.

3. Provide a list of the main advantages and disadvantages of *SIMD* and *MIMD* machines.

4. Provide a list of the main advantages and disadvantages of Shared-memory and Message-passing paradigm.

5. List three engineering applications, which you are familiar with, for which SIMD is most efficient to use, and another three for which MIMD is most efficient to use.

6. Consider the case of connecting N processors and N memory modules using each of the interconnection networks shown in Fig. 11.1. Assume that T is the time required for a processor to access an item in a memory module and that all processors make a request to access distinct memory module. Compute the worst-case possible delay expected in each of the four interconnection networks.

7. It was mentioned that a given SIMD machine could be characterized using a 5-tuple (N, C, I, M, F). You are required to select three different recent SIMD machines and provide in a tabular form each of the 5-tuples that characterizes them.

8. Assume that a simple addition of two elements requires a unit time. You are required to compute the execution time needed to perform the addition of a 40×40 elements array using each of the following arrangements:

   (a) An SIMD system having 64 processing elements connected in nearest-neighbor fashion. Consider that each processor has only his local memory.
   (b) An SIMD system having 64 processing elements connected to a shared memory through an interconnection network. Ignore the communication time.
   (c) An MIMD computer system having 64 independent elements accessing a shared memory through an interconnection network. Ignore the communication time.
   (d) Repeat b and c above if the communication time takes two time units.

9. Provide a concise discussion on the suitability of each of the four attributes of interconnection networks (mode of operation, control strategy, switching mechanism, and topology) for each of the four different interconnection networks shown in Fig. 11.1. Make sure that you justify the suitability of a given attribute to a given interconnection network.

10. Consider the case of a multiprocessor system consisting of *N* processors. Assume that the time needed for each processor to execute a given critical section is *t* and that *f* represents the fraction of operations which can be parallelized. Assume also that a single processor will need a time *T* to execute the same task. Show that the total execution time using *N* processors is given by $T_N = (1 - f) \times T + \frac{f \times T}{N} + t$.
What is the number of processors, *N,* needed in order to minimize the total execution time T_N.